

**HIGHLY AVAILABLE, MONOTONIC INCREASING SEQUENCE NUMBER
GENERATION**

Michael R. Krause
12523 Bear Creek Road
Boulder Creek, California 95006
Citizenship: U.S.A.

Kimberly K. Scott
2639 Meta Drive
San Jose, California 95130
Citizenship: U.S.A.

Fred B. Worley
750 Harvard Avenue, #1
Santa Clara, California 95051
Citizenship: U.S.A.

TECHNICAL FIELD

The present invention relates to monotonic sequence number generation for use by distributed data processing and in particular to monotonic sequence number generation for use by distributed database-related applications in a highly available and low latency manner.

BACKGROUND

In present business environments, many information systems exist to process transaction information. For example, certain banking information systems may utilize transaction information to track transactions initiated by account holders. A banking information system may store transaction information in a database. Furthermore, the banking information system may utilize transaction numbers to provide order to database actions. The banking information system may assign an ever-increasing (monotonic) sequence of numbers to database actions, so that their order may be retained and processing of the actions may occur in an ordered manner. For example, the banking information system may utilize transaction numbers to ensure that an electronic transfer of funds does not occur before a deposit transaction is completed.

The concept of utilizing transaction numbers is clearly applicable to a very large range of data processing applications in addition to banking information systems. For example, many business to business information systems may require sequence numbering to manage purchase orders, delivery arrangements, invoicing, and payment information via extensive databases. Broadly speaking, the provision of sequence numbers may be thought of as a form of time stamping. Of course, there are numerous applications of time stamping in many industries. Accordingly, it shall be further appreciated that the sequence numbering process is applicable to many applications beyond applications related to financial or commercial environments. In general, any application that orders a significant amount of data objects may utilize sequence number generation. Distributed applications especially benefit from sequence number generation.

Sequence numbering is a relatively simply process to implement upon a single processor system. However, it is evident that most practical applications of sequence numbers require scalability, i.e. the sequence numbering may be employed up to an arbitrary number of logical nodes which are usually distributed across a communication medium or system fabric. To achieve scalability, various mechanisms may be employed, such as

determining which process maintains the current sequence number; which process is incrementing the number; which process is retaining a master list of sequence numbers; and the like. To implement these tasks in a distributed manner in a multi-node environment, the respective processors located at the various nodes allocate a portion of processing time to threads responsible for these overhead mechanisms. Additionally, an amount of signaling occurs over the communication medium for similar allocation to overhead mechanisms. When these mechanisms are implemented in software, the overhead processing and communication protocol signaling are significant. By allocating a significant portion of system resources to the overhead tasks, milliseconds of delay are introduced which significantly reduces the scalability of the system. In fact, delays of several milliseconds may cause negative scaling, i.e. coordinating the functionality over several nodes may actually cause a reduction in processing capacity.

Similarly, hardware implementations of the monotonic sequence number generation proved problematic. Hardware implementations have involved placing a device in a system fabric accessible by the various processors by interfaces. In this environment, latency is reduced. However, the price of reducing latency in this implementation is reduced availability. Specifically, this implementation creates a single point of failure that may cause the entire system to malfunction.

SUMMARY OF THE INVENTION

100-05150-01

5 The present invention is directed to a system and method which provides a monotonic sequence number generation that is highly available and possesses minimal latency. The architecture of the present invention involves establishing at least a primary generator and a secondary generator. Applications that utilize transaction numbering to order data may initiate a call in software to obtain a sequence number. The software call utilizes an interface to bypass ordinary network or communication channel protocols. The interface, defined by a hardware or firmware implementation device, preferably generates a sequence number request to the primary unit. After receiving the sequence number request, the primary sequence number generator determines a next sequence number in a monotonic manner. The primary sequence number updates its memory to reflect that the determined sequence number has been utilized. The primary sequence number generator forwards the determined sequence number to the secondary sequence number generator. The secondary sequence number generator updates its memory to reflect that the determined sequence number has been utilized. The secondary sequence number generator forwards the determined sequence number to the initial hardware implementation device. Finally, the hardware implementation device returns the sequence number to the appropriate application process.

20 It is a further aspect of the present invention to provide sequence number generation in a manner that is robust against failure. Since the secondary generator forwards the determined sequence number to the hardware implementation device, the secondary generator maintains a record of the last sequence number utilized by an application. Occasionally, the primary unit may become unavailable due to hardware difficulties or communication link failure. If this occurs, the underlying architecture may readjust by reassigning the secondary generator as the primary generator. Similarly, the underlying architecture may assign a new secondary generator. By doing so, the current sequence number is not lost upon failure of the primary sequence number generator. Future sequence requests may be forwarded to the new

25

primary generator, thereby increasing availability. Secondly, the architecture may perform the adaptation in a manner that is transparent to application processes.

The foregoing has outlined rather broadly the features and technical advantages of the present invention in order that the detailed description of the invention that follows may be better understood. Additional features and advantages of the invention will be described hereinafter which form the subject of the claims of the invention. It should be appreciated by those skilled in the art that the conception and specific embodiment disclosed may be readily utilized as a basis for modifying or designing other structures for carrying out the same purposes of the present invention. It should also be realized by those skilled in the art that such equivalent constructions do not depart from the spirit and scope of the invention as set forth in the appended claims.

10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194

BRIEF DESCRIPTION OF THE DRAWING

For a more complete understanding of the present invention, reference is now made to the following descriptions taken in conjunction with the accompanying drawing, in which:

FIGURE 1 comprises a simplified system architecture and signal flow to illustrate the present invention;

5 FIGURE 2 comprises a redundant system architecture to illustrate the present invention;

FIGURE 3 illustrates processing steps associated with initialization of a highly available sequence number generation architecture;

10 FIGURE 4 illustrates a light-weight call by an application process to obtain a sequence number and associated underlying processing steps; and

FIGURE 5 depicts a block diagram of a computer system which is adapted to use the present invention.

10001459-1

DETAILED DESCRIPTION

Turning now to the drawing, FIGURE 1 illustrates a simplified signaling progression associated with sequence number generation. An application process at node C may generate a sequence number (SN) request. Firmware at node C generates a SN message with the ID of node A thereby causing the request to be routed to node A. Node A responds to the request by creating an SN response comprising the ID of node B, thereby causing the SN response to be routed to node B. After receipt, node B forwards the request comprising the ID of node C, thereby routing the SN response to its ultimate destination, node C. The firmware initiates a direct memory access (DMA) to place the received response in a memory location accessible by the originating application process.

It is an advantage of the present invention to facilitate the embodiment of sequence number generation in a wide variety of hardware and/or firmware implementations. By way of example and without limitation, the sequence numbering in the node system of FIGURE 1 may be implemented by adapter cards located at nodes A, B, and C. The implementation of the present invention is not limited to adapter cards. For example, the present invention may be implemented in host/appliance software, firmware, ASICs, fabric elements (e.g. augmenting a switch implementation) or any combination thereof. It shall be appreciated that any further discussion of adapter cards or firmware refers broadly to all of the preceding implementations.

In FIGURE 1, the adapter cards may be employed to provide replicated fabrics. In this exemplary embodiment, each node may comprise two cards. In normal operation, node C forwards a request to node A. Node A may increment the present value, DMA the incremented value to a memory location in its host processor system, and forward the value to node B. Node B may DMA the value to a memory location in its host processor system and forward the result to C. If the operating card on node A becomes inaccessible for some reason (card malfunction, switch failure, cable failure, and/or the like), failure over to the other card may occur instead of changing the primary and secondary designations. Since the

incremented value is stored in the memory of the host systems, it is sufficient to reinitialize to the other cards. Accordingly, this embodiment provides an alternative recovery mechanism. However, it shall be appreciated that this configuration permits recovery either through DMA operations or failure to the other node. It is not necessary to restrict recovery to only one of these mechanisms.

The SN representation may comprise an arbitrarily large number of bits. In a preferred embodiment, the SN representation comprises 48-bits. Obviously, the sequence numbering space associated with a bit-limited representation is finite. Accordingly, the sequence number generation is not monotonic in a strict mathematical sense. However, it is estimated that normal database operation in connection with 64-bit sequence numbering would require years to exhaust the attendant sequence numbering space. Accordingly, the 48-bit or 64-bit embodiments are sufficient to establish the desired ordering functionality.

The present invention preferably utilizes pipelining to deliver sequence number requests. Pipelining involves a distinction between strong ordering and weak ordering. In strong ordering, sequence numbers are assigned monotonically in response to the arrival of sequence number requests. Additionally, strong ordering involves delivering the sequence numbers in the same order as the sequence number request arrivals. In weak ordering, sequence numbers are also assigned monotonically in response to the arrival of sequence number requests. However, weak ordering does not require that the sequence number response values be delivered in order according to the request arrivals. Pipelining may be utilized if node C comprises numerous processors. By utilizing weak ordering, pipelining allows numerous sequence number requests to be processed concurrently without having node C wait until individual requests are completed before initializing another request. Additionally or alternatively, a transaction identifier may be included in each sequence request packet. By utilizing such transaction identifiers, it may be possible to have potentially thousands of outstanding requests at any one time. It is a further advantage of the present inventive system and method to achieve greater scalability by utilizing pipelining.

Nonetheless, it shall be appreciated that the present invention may utilize strong ordering techniques.

FIGURE 2 illustrates an exemplary architecture employing the present invention. The network of FIGURE 2 comprises a plurality of nodes (201-208) and a plurality of switches (209A and 209B). It shall be appreciated that the architecture is merely exemplary and that an arbitrary number of nodes and switches may be utilized. Each node comprises a plurality of HW cards (HW cards 0 and HW cards 1) to facilitate connection to nodes 209A and 209B, respectively. The HW cards preferably provide an interface to the monotonic sequence generation fabric. The cards preferably provide lower level routines to request sequence numbers, issue or store sequence numbers for generator units, and/or initiate recovery measures. HW cards 0 of nodes 201-208 define logical connections from each node to switch 209A. HW cards 1 of nodes 201-208 define logical connections from each node to switch 209B. Accordingly, this embodiment employs a duplicated fabric configuration.

Additionally, it shall be appreciated that a requesting node may be one of the primary or secondary generators. Utilizing the preceding discussed configuration, if an application upon node 204 requests a sequence number, HW card 0 of node 201 may increment the current value, DMA the value, and forward the value to HW card 0 of node 203. Node 203 may increment the value and forward the value back to HW card 0 of node 204. In this manner, it is ensured that the current value is retained on both the primary and secondary generators. After receipt of the return value, HW card 0 may DMA the value for receipt by the appropriate application process.

By way of example, the system may be initially configured to have node 204 as the primary generator and node 203 as the secondary generator. If node 201 issues a sequence number request and the system is utilizing the first fabric, node 201 may transmit the request via its HW card 0 through switch 209A. After receipt of the request, HW card 0 of node 203 may perform the increment operation, DMA operation, and forward operation through switch 209A. After receipt of the forwarded value, HW card 0 of node 204 may perform the DMA

operation and the forward operation through switch 209A to HW card 0 of node 201. After receipt of the value, HW card 0 of node 201 may DMA the value for receipt by the appropriate application process. Upon failure (for example the failure of HW card 0 of node 208), the system may be reconfigured by system management logic to utilize the secondary fabric defined by HW cards 1 and switch 209A.

Alternatively, if both cards of the primary generator (node 204) fail, the manager process may detect the failure. The manager process may cause the secondary generator (node 203) to become the new primary generator, since node 203 has the current sequence number value. The manager process may preferably be implemented as middleware software that is independent of the applications utilizing the sequence number generation. Secondly, the manager process may select a new secondary generator. The new secondary generator receives the current value from node 203. Of course, the manager process communicates the identity of the new primary and secondary generators to the various nodes. At this point, all of the nodes except for node 204 are able to communicate with the new primary and secondary generators. Accordingly, process applications on these nodes are able to continue functions, since the sequence number generation remains available. Accordingly, the failure of both cards of node 204 only causes node 204 to cease to function. Thus, the present invention avoids a single point of failure, since failure of one node does not impede the remaining nodes of the network.

It shall be appreciated that, in the event of unavailability of the secondary generator, identical processing may be utilized to select an alternative secondary generator and to provide the current sequence number to the newly designated secondary generator. It shall be further appreciated that the preceding architecture is merely one embodiment of the present invention. The present invention does not require any rigidly defined hardware implementation. The present invention is preferably implemented transparently within an existing network or fabric. For example, it is contemplated that the network of FIGURE 1 could constitute a pre-existing configuration. The present invention may be employed upon

the pre-existing configuration with minimal modification by adding the sequence number generation mechanisms with connections to pre-existing switches 209A and 209B. Accordingly, it is an advantage of the present invention to provide a highly available sequence number generation to a system of nodes with minimal modifications.

5 As previously noted, the present invention preferably employs SN interfaces to provide sequence number generation to application processes in a transparent manner. Preferably, a library of user-space functions/routines is provided to access lower level sequence number routines of a selected device hardware/firmware implementation. The library may comprise the following exemplary functions/routines:

10 snbgn() - a function that is responsible for seeding the SN generator. Specifically, the seed operation selects a current sequence number that is sufficiently greater than a last known value so that it is highly probably that the current sequence number is beyond any value currently in flight. The routine may open a domain socket to a management thread and send a message containing the new SN value. The thread may contact the primary and secondary
15 generators to determine if they are operational and may "seed" the firmware.

snnew() - a function that is utilized by process applications to transparently generate a new SN value. The function call operates from the perspective of an application as an ordinary function call. The function may preferably be implemented as a light-weight call, as will be discussed in greater detail below. In the event of a time-out, the function may invoke a
20 heavy-weight system call which may invoke sleeping.

sncur() - a function that returns the current global SN value.

snadj() - a function that allows the current global SN value to be incremented by an offset to insure that no sequence number replication occur after a recovery is completed.

It shall be appreciated that the use of a light weight system call such as `snnew()` is only one possible implementation. For example, similar functionality may be achieved utilizing a work request to the underlying hardware without requiring an application to perform a system call. Such an implementation would define a hardware work request which provides the same set of services and essentially the same output would occur. Alternatively, a system call may be implemented via a user application library call utilizing operating system bypass technology. Accordingly, it shall be appreciated that terms system call, light weight system call, and user application library call shall be interpreted equivalently.

As previously noted, the sequence number generation may be employed in a wide variety of hardware configurations. Additionally, the configuration within a specific network or system may vary due to installation or removal of new resources, links, and/or the like. Accordingly, the present invention may employ a configuration process to establish primary and secondary generators. FIGURE 3 illustrates exemplary steps in such a configuration process. The configuration process may be initiated by a call to `snbgn()`. The function `snbgn()` is preferably implemented so that it communicates a seed message to a node management thread, instead of the local hardware/driver. In step 302, the node management thread may then initiate a primary and secondary generator election process. In step 303, the node management communicates a connectivity message to determine the status of all nodes within a cluster. The node management thread may then select a primary generator. After selecting the primary generator, the node management thread attempts a seed operation in step 304. Similarly, the node management thread may select a secondary generator and attempt a second seed operation in step 305. Of course, if any of the seed operations fail, the node management thread may select another generator and repeat the respective seeding step as many times as necessary to complete the selection process. After the primary and secondary generators have been selected and seeded, the ID of each generator may be broadcast to all connected nodes.

FIGURE 4 illustrates an exemplary SN operation within a light-weight system call. In step 401, the processing steps may begin by an application process performing a `snnew()` call. In step 402, the global state is examined. If the state in the firmware device is enabled and the fabric is viable, the process continues in step 403a. It shall be appreciated that the implementation utilizing firmware is only one embodiment of the present invention. The steps associated with firmware in this example could also be implemented in a variety of hardware structures. In step 403b, the global state is determined to be deficient and a return error message is generated for recovery purposes. In step 404, the operating processor is determined. The processor completion flag is cleared (step 405) and a PIO is performed to request a SN value from the fabric (step 406). Firmware sends an SN request comprising the primary ID to the primary generator (step 406-1). The primary generator receives the message, increments local value, and forwards a response to the secondary generator (step 406-2). The secondary generator receives the message, records the SN value, and forwards the response to the initiating firmware (406-3). Firmware performs a DMA to place the value into memory and sets the processor completion flag (step 406-4). Concurrently to the underlying operations of steps 406-1 through 406-4, a spin loop is executed until the completion flag is set or the maximum spin count is exceeded (step 407). Upon detection of the completion flag, the processor breaks out of its spin loop and returns the value to the application process that performed the `snnew()` call. If the processor breaks from the spin loop after completing the maximum spin count, a heavy-weight call may be initiated as will be discussed in greater detail below.

It shall be appreciated that the preceding steps facilitate superior data processing performance. Specifically, the hardware implementation of the SN request process allows a SN response value to be received in microseconds. By minimizing the latency of the system, it is not necessary to perform a sleep operation or perform a context switch out. The present invention avoids wasting processing cycles to store and retrieve data for caches associated with placing the calling application process in a sleep state and subsequently reawakening the process.

As mentioned earlier, the present invention may utilize a heavy-weight call in the rare event that a light-weight call is unsuccessful in obtaining a sequence number in a predetermined amount of time. In a heavy-weight call, the application process that made the initial sequence number call is placed into a sleep state. Accordingly, associated cache data is stored and retrieved as other processes utilize processor cycles during the sleep operation. Also, the heavy-weight call may notify the sequence number management thread that a problem has been encountered and recovery measures should be initiated. The management thread may attempt to determine the cause of the delay, such as a malfunctioning card, line, and/or the like and reconfigure accordingly. If the recovery is successful, the management thread may communicate successful recovery to the heavy-weight call process. The heavy-weight call process may then re-request a sequence number. Upon receipt of the sequence number, the heavy-weight call may cause the application process to awaken from the sleep operation and communicate the received sequence number to the process. If the management process is unable to initiate a recovery, the heavy-weight call may communicate an appropriate error message to the calling application process.

When implemented in software, certain elements of the present invention are essentially the code segments to perform the necessary tasks. The program or code segments can be stored in a processor readable medium or transmitted by a computer data signal embodied in a carrier wave, or a signal modulated by a carrier, over a transmission medium. The "processor readable medium" may include any medium that can store or transfer information. Examples of the processor readable medium include an electronic circuit, a semiconductor memory device, a ROM, a flash memory, an erasable ROM (EROM), a floppy diskette, a compact disk CD-ROM, an optical disk, a hard disk, a fiber optic medium, a radio frequency (RF) link, etc. The computer data signal may include any signal that can propagate over a transmission medium such as electronic network channels, optical fibers, air, electromagnetic, RF links, etc. The code segments may be downloaded via computer networks such as the Internet, Intranet, etc.

FIGURE 5 illustrates computer system 500 adapted to use the present invention. Central processing unit (CPU) 501 is coupled to system bus 502. The CPU 501 may be any general purpose CPU such as an Intel Pentium processor. However, the present invention is not restricted by the architecture of CPU 501 as long as CPU 501 supports the inventive operations as described herein. Bus 502 is coupled to random access memory (RAM) 503, which may be SRAM, DRAM, or SDRAM. ROM 504 is also coupled to bus 502, which may be PROM, EPROM, or EEPROM. RAM 503 and ROM 504 hold user and system data and programs as is well known in the art.

Bus 502 is also coupled to input/output (I/O) controller card 505, communications adapter card 511, user interface card 508, and display card 509. The I/O card 505 connects to storage devices 506, such as one or more of hard drive, CD drive, floppy disk drive, tape drive, to the computer system. Communications card 511 is adapted to couple the computer system 500 to a network 512, which may be one or more of telephone network, local (LAN) and/or wide-area (WAN) network, Ethernet network, and/or Internet network. User interface card 508 couples user input devices, such as keyboard 513 and pointing device 507, to the computer system 500. The display card 509 is driven by CPU 501 to control the display on display device 510.

Although the present invention and its advantages have been described in detail, it should be understood that various changes, substitutions and alterations can be made herein without departing from the spirit and scope of the invention as defined by the appended claims. Moreover, the scope of the present application is not intended to be limited to the particular embodiments of the process, machine, manufacture, composition of matter, means, methods and steps described in the specification. As one of ordinary skill in the art will readily appreciate from the disclosure of the present invention, processes, machines, manufacture, compositions of matter, means, methods, or steps, presently existing or later to be developed that perform substantially the same function or achieve substantially the same result as the corresponding embodiments described herein may be utilized according to the

present invention. Accordingly, the appended claims are intended to include within their scope such processes, machines, manufacture, compositions of matter, means, methods, or steps.

FOR FEED: 65 FEB 60